Towards a Unified Framework for Fair and Stable Graph Representation Learning

Chirag Agarwal[1] **Himabindu Lakkaraju**[*1] **Marinka Zitnik**[*1]

[1]Harvard University

Abstract

As the representations output by Grap
works (GNNs) are increasingly emp
world applications, it becomes import
that these representations are fair a
this work, we establish a key connec
counterfactual fairness and stability
it to propose a novel framework, NIF
Fairness and stabiliTY), which can
any GNN to learn fair and stable rep
We introduce a novel objective functi
taneously accounts for fairness and
develop a layer-wise weight normali
the Lipschitz constant to enhance ne
passing in GNNs. In doing so, we
ness and stability both in the objective function as well as in the GNN architecture. Further, we show theoretically that our layer-wise weight normalization promotes counterfactual fairness and stability in the resulting representations. We introduce three new graph datasets comprising of high-stakes decisions in criminal justice and financial lending domains. Extensive experimentation with the above datasets demonstrates the efficacy of our framework.

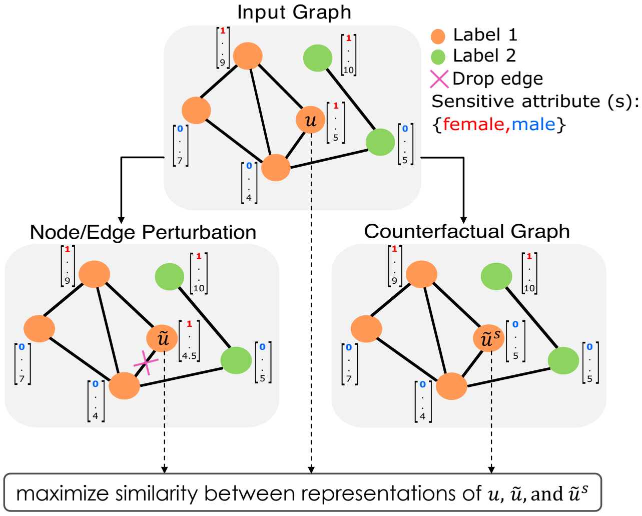


Figure 1: Our framework NIFTY can learn node representations that are both fair and stable (*i.e.,* invariant to the sensitive attribute value and perturbations to the graph structure and non-sensitive attributes) by maximizing the similarity between representations from diverse augmented graphs.

1 INTRODUCTION

Over the past decade, there has been a surge of interest in leveraging GNNs for graph representation learning. GNNs have been used to learn powerful representations that enabled critical predictions in downstream applications—*e.g.,* predicting protein-protein interactions [Gainza et al., 2020, Huang et al., 2020], drug repurposing [Gysi et al., 2020, Zitnik et al., 2018], crime forecasting [Jin et al., 2020], news and product recommendations [Ying et al., 2018]. As GNNs are increasingly implemented in real-world applications, it becomes important to ensure that these models and the resulting representations are safe and reliable. More specifically, it is important to ensure that these models and the representations they produce are not perpetrating undesirable discriminatory biases (*i.e.,* they are fair), and are also robust to attacks resulting from small perturbations to the graph structure and node attributes (*i.e.,* they are stable).

A myriad of GNN methods with various neighborhood aggregation schemes have recently been developed (*e.g.,* Kipf and Welling [2017], Hamilton et al. [2017], Xu et al. [2018, 2019], Veličković et al. [2019]). While these methods achieve state-of-the-art performance in tasks such as node classification and link prediction, these methods can be prone to discrimination and instability [Dai and Wang, 2021, Rahman et al., 2019, Bose and Hamilton, 2019]. Fur-

[*]Equal Contribution

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

thermore, prior work has argued that GNNs not only capture the undesirable biases prevalent in the data, but may also exacerbate them thanks to their message passing schemes [Dai and Wang, 2021]. Generally, in graphs such as social networks, nodes with similar sensitive attribute (*e.g.,* race, age) values are likely to connect to each other [Dai and Wang, 2021]. Since GNNs compute node representations by propagating and aggregating neural messages along edges in graph neighborhoods, nodes with similar sensitive attribute values are likely to share similar representations leading to severe discriminatory biases, *i.e.,* downstream predictions may be highly correlated with sensitive attributes.

Recent research has treated fairness and stability in GNNs as independent problems and proposed standalone solutions for the same. For example, Dai and Wang [2021] proposed FairGNN to promote fairness in GNNs through an objective function that incorporates group fairness measures such as statistical parity and equality of opportunity. On the other hand, Zhu et al. [2019] aimed to make GNNs stable and robust to adversarial attacks. While these techniques provide a promising approach to study fairness and stability independently, it remains an open question whether there are any deeper connections between fairness and stability in GNNs, and if these properties can be achieved simultaneously.

Present work. Here, we address the problem of learning node representations that are both fair and stable. To tackle this problem, we first identify a key connection between counterfactual fairness and stability. While stability accounts for robustness w.r.t. small random perturbations to node attributes and/or edges, counterfactual fairness accounts for robustness w.r.t. modifications of the sensitive attribute. We leverage this connection to propose a novel framework, NIFTY (uNIfying Fairness and stabiliTY), that can be used with any existing GNN model to learn fair and stable representations. Our framework exploits the aforementioned connection to enforce fairness and stability both in the objective function as well as in the GNN architecture. More specifically, we introduce a novel objective function which simultaneously optimizes for counterfactual fairness and stability by maximizing the similarity between representations of the original nodes in the graph, and their counterparts in the augmented graph (Fig. 1). Nodes in the augmented graph are generated by slightly perturbing the original node attributes and edges or by considering counterfactuals of the original nodes where the value of the sensitive attribute is modified. We also develop a novel method for improving neural message passing by carrying out layer-wise weight normalization using the Lipschitz constant. We theoretically show that this normalization promotes counterfactual fairness and stability of learned representations. To the best of our knowledge, this work is the first to tackle the problem of learning node representations that are both fair and stable.

We introduce and experiment with three new graph datasets comprising of critical decisions in criminal justice (if a defendant should be released on bail) and financial lending (if an individual should be given loan) domains. Our results show that NIFTY improves the fairness and stability of five GNNs by 92.01% and 60.87% respectively (on an average) without sacrificing predictive performance. We also observe that the resulting representations become fairer not only w.r.t. the notion of counterfactual fairness but also w.r.t. other notions of group fairness such as statistical parity and equality of opportunity. Further, our results establish that enforcing fairness and stability both in the objective function as well as in the GNN architecture can be incredibly beneficial for learning fair and stable representations.

2 RELATED WORK

This work lies at the intersection of fairness and stability in machine learning, and Graph Neural Networks (GNNs). Below we discuss related work for each of these topics.

Fairness. Several competing and contrasting notions of fairness have been proposed in recent literature. They can be broadly categorized into: 1) *group fairness*, which emphasizes that minority groups should receive similar treatment as that of advantaged groups [Berk et al., 2018, Hardt et al., 2016], 2) *individual fairness*, which requires that *similar* individuals should be treated similarly [Dwork et al., 2012], and 3) *counterfactual fairness*, which captures the intuition that a decision pertaining to an individual is fair if changing the individual's sensitive attribute value does not affect the decision [Kusner et al., 2017]. Furthermore, various metrics have been proposed to realize each of the aforementioned notions of fairness. For example, statistical (demographic) parity, equalized odds, equality of opportunity, and predictive parity are metrics proposed to enforce group fairness. These metrics have also been leveraged to develop new objective functions for constructing machine learning models that are both fair and accurate [Zafar et al., 2017b,a]. Prior research has also established that certain notions of fairness (calibration and balance conditions) are fundamentally incompatible and cannot be simultaneously optimized [Kleinberg et al., 2017, Chouldechova, 2017].

Graph Neural Networks. Deep learning on graphs and GNNs, in particular, learn how to represent nodes in a graph as points, *i.e.,* embeddings, in a vector embedding space, where the geometry of the embedding space is optimized to reflect topology of the graph as well as node attribute information [Wu et al., 2020]. Motivated by spectral graph convolutions [Hammond et al., 2011, Defferrard et al., 2016], Graph Convolutional Networks (GCN) [Kipf and Welling, 2017] specified deep transformation functions akin to applying convolutional filters over local graph neighborhoods. The subsequent methods, *e.g.,* Gilmer et al. [2017], Hamilton et al. [2017], Hu et al. [2020], Lee et al. [2019], Alsentzer et al. [2020], developed efficient algorithms for rich types of graphs and larger structures, including edges, subgraphs, and entire graphs by generating embeddings through a series of transformations that exchange embeddings between neighboring nodes in the graph. For example, Jumping Knowledge (JK) Networks [Xu et al., 2018] use skip connections to leverage diverse local neighborhoods and generate richer representations. Similarly, Graph Isomorphism Networks (GIN) [Xu et al., 2019] adaptively adjust the importance weights of nodes and Deep Graph Infomax (DGI) [Veličković et al., 2019] relies on maximizing mutual information between patch representations and high-level graph summaries to produce node representations.

Fairness and Stability in GNNs. Recent studies addressed the issues of fairness and stability in GNNs [Dai and Wang, 2021, Fisher et al., 2020, Geisler et al., 2020, Bose and Hamilton, 2019, Rahman et al., 2019, Zhu et al., 2019, Zhang and Zitnik, 2020]. To achieve fairness, existing work de-biases embeddings with respect to sensitive attributes via adversarial learning frameworks [Dai and Wang, 2021, Bose and Hamilton, 2019]. These methods use regularization to implement the notion of group fairness; however, they are incapable of achieving counterfactual fairness. To achieve stability, recent methods use adversarial training [Zügner and Günnemann, 2019], robust message-aggregation [Geisler et al., 2020], and attention mechanisms [Zhu et al., 2019] to defend GNNs against a variety of attacks that perturb discrete graph structure or node attributes. In contrast, our unifying framework can learn graph embeddings that are simultaneously fair and stable.

3 PRELIMINARIES

Notation. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ denote an undirected graph comprising of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ denote the set of node attribute vectors corresponding to all the nodes in $\mathcal{V}$. More specifically, $\mathbf{x}_v \in \mathbf{X}$ is an M-dimensional vector which captures the attribute values of node $v \in \mathcal{V}$. Let $N = |\mathcal{V}|$ denote the number of nodes in the graph and let $\mathbf{A} \in \mathbb{R}^{N \times N}$ be the graph adjacency matrix where element $\mathbf{A}_{uv} = 1$ if there exists some edge $e \in \mathcal{E}$ between nodes u and v, and $\mathbf{A}_{uv} = 0$ otherwise. We also use $\mathcal{N}_u$ to denote the set of immediate neighbors of node u, *i.e.,* $\mathcal{N}_u = \{v \in \mathcal{V} | A_{uv} = 1\}$. Furthermore, let $\mathbf{I}_u \in \{0, 1\}^N$ denote the binary incidence vector which captures all the edges incident on node u, *i.e.,* $\mathbf{I}_{uv} = 1$ if an edge exists between nodes u and v otherwise it is set to 0. Finally, we introduce $\mathbf{b}_u$ to capture all the information associated with node u, *i.e.,* $\mathbf{b}_u = [\mathbf{x}_u; \mathbf{I}_u]$ denotes the concatenation of node attribute vector and binary incidence vector corresponding to node u. We also generate an augmented graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}', \tilde{\mathbf{X}})$ as follows: for each node $u \in \mathcal{V}$ in the original graph, we generate a corresponding node in the augmented graph by slightly perturbing the attribute values, incident edges, and/or modifying the value of the sensitive attribute of node u. The adjacency matrix and node attribute vectors corresponding to this augmented graph $\mathcal{G}'$ are denoted by $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{X}}$.

We consider a GNN with K layers and denote the representations output by each of these layers as $\mathbf{h}_u^1, \mathbf{h}_u^2, \cdots \mathbf{h}_u^{K-1}, \mathbf{h}_u^K$ for a given node u. We use $\mathbf{z}_u$ to denote the representation output by the last layer of the GNN for node u *i.e.,* $\mathbf{z}_u = \mathbf{h}_u^K$. Analogously, $\tilde{\mathbf{z}}_u$ denotes the representation output by the last layer of the GNN for node u in the augmented graph $\mathcal{G}'$. We assume that the (dis)similarity between any two node representations is given by a distance metric $D : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Our goal is to learn an encoder function $\textsc{Enc}$ which maps a given node u to a representation $\mathbf{z}_u$ *i.e.,* $\textsc{Enc}(u) = \mathbf{z}_u$. Lastly, let f denote a downstream classifier that maps the node representation $\mathbf{z}_u$ of a given node u to a class label $\hat{y}_u$.

Graph Neural Networks. Many GNNs can be formulated as message passing networks [Wu et al., 2020] specified by trainable operators $\textsc{Msg}$, $\textsc{Agg}$, and $\textsc{Upd}$. In a K-layer GNN, the operators are recursively applied on $\mathcal{G}$, specifying how neural messages (*i.e.,* embeddings) are exchanged between nodes, aggregated, and transformed to arrive at final node representations in the last layer of transformations. Typically, a message between a pair of nodes (u, v) in layer k is defined as a function of hidden representations of nodes $\mathbf{h}_u^{k-1}$ and $\mathbf{h}_v^{k-1}$ from the previous layer: $\mathbf{m}_{uv}^k = \textsc{Msg}(\mathbf{h}_u^{k-1}, \mathbf{h}_v^{k-1})$. In $\textsc{Agg}$, messages from $\mathcal{N}_u$ are aggregated as $\mathbf{m}_u^k = \textsc{Agg}(\mathbf{m}_{uv}^k | u \in \mathcal{N}_u)$. In $\textsc{Upd}$, the aggregated message $\mathbf{m}_u^k$ is combined with $\mathbf{h}_u^{k-1}$ to produce u's representation for layer k as $\mathbf{h}_u^k = \textsc{Upd}(\mathbf{m}_u^k, \mathbf{h}_u^{k-1})$. Final node representation $\mathbf{z}_u = \mathbf{h}_u^K$ is the output of the last layer.

Fairness and Stability. Our goal is to learn node representations that are fair and stable. More specifically, the notions of fairness and stability that we consider in this work are counterfactual fairness and Lipschitz continuity respectively. Below, we provide definitions of these notions and formalize them in the context of graph representation learning.

Counterfactual Fairness: A function is considered to be counterfactually fair if its output is independent of the sensitive attribute, *i.e.,* changing the sensitive attribute value of any given instance should not affect the output of the function for that instance. In the context of graph representation learning, this notion can be interpreted as follows: node representations output by encoders should be independent of the sensitive attribute.

Definition 1. *An encoder function* $\textsc{Enc}$ *satisfies counterfactual fairness if the following holds for any given node* u*:*

$$\textsc{Enc}(u) = \textsc{Enc}(\tilde{u}^s) \tag{1}$$

where $\tilde{u}^s$ is a node in the augmented graph which is generated by modifying/flipping the value of the sensitive attribute (s) of node u while keeping everything else constant.

Stability via Lipschitz Continuity: A function is considered to be stable according to the notion of Lipschitz continuity if slightly perturbing any given instance does not drastically change the output of the function. In the context of graph representation learning, this notion can be interpreted as follows: small perturbations to node attributes and/or incident edges should not drastically change the resulting representations.

Definition 2. *An encoder function* ENC *is stable according to the notion of Lipschitz continuity if:*

$$||\text{ENC}(\tilde{u}) - \text{ENC}(u)||_p \leq L||\tilde{\mathbf{b}}_u - \mathbf{b}_u||_p, \qquad (2)$$

where $\tilde{u}$ is a node in the augmented graph generated by perturbing u's attribute values and/or incident edges, $\mathbf{b}_u$ and $\tilde{\mathbf{b}}_u$ capture the attribute and incident edge information for nodes u and $\tilde{u}$ respectively, and L is the Lipschitz constant.

4 OUR FRAMEWORK NIFTY

Next, we describe our framework NIFTY which aims to generate fair and stable graph embeddings. To achieve this goal, NIFTY infuses fairness and stability in the objective function (Section 4.1) as well as in the architecture (Section 4.2) of underlying GNN.

Problem formulation (Fair and Stable embeddings). *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$,* NIFTY *aims to generate d-dim. embeddings $\mathbf{z}_u \in \mathbb{R}^d$ that are counterfactually fair (Eq. 1) and stable to attribute and structural perturbations of $\mathcal{G}$ (Eq. 2).*

4.1 ENFORCING FAIRNESS AND STABILITY IN THE OBJECTIVE FUNCTION

To infuse fairness and stability in the objective function, we introduce a triplet-based objective that maximizes the agreement between the original graph and its counterfactual and noisy views. To this end, we build off the Siamese networks to maximize this agreement, *i.e.,* the two augmented network neighborhoods and the augmented attribute vectors of the same node should result in the same embedding [Chen et al., 2020, Chen and He, 2020]. Next, we describe the graph augmentation procedure.

Generating augmented views of graph structure and attribute information is key for the Siamese learning approach. We generate them using node-, sensitive attribute-, and edge-level perturbations.

a) Perturbing node attributes. We draw a random attribute masking vector $\mathbf{r} \in \{0,1\}^M$ from a Bernoulli distribution, *i.e.,* $\mathbf{r} \sim \mathcal{B}(p_n)$, where p_n is the probability of independently perturbing each attribute (except for the sensitive attribute s) in $\mathbf{x}_u$. The augmented attribute vector is then defined as $\tilde{\mathbf{x}}_u = \mathbf{x}_u + \mathbf{r} \circ \delta$, where $\delta \in \mathbb{R}^M$ is sampled from a normal distribution.

b) Counterfactual perturbation of sensitive attribute. We modify the value of sensitive attribute s in $\mathbf{x}_u$ to generate a counterfactual. More specifically, we consider the case where the sensitive attribute is a binary variable (i.e., $s \in \{0,1\}$) and we create a counterfactual node $\tilde{u}^s$ by flipping the value of s from 0 to 1 or vice-versa.

c) Perturbing graph structure. We draw a random binary mask from a Bernoulli distribution, *i.e.,* $\mathbf{R}_e \sim \mathcal{B}(1-p_e)$, where $\mathbf{R}_e \in \{0,1\}^{N \times N}$ and p_e denotes the probability with which an edge is dropped from $\mathcal{G}$. We construct the augmented adjacency matrix as $\tilde{\mathbf{A}} = \mathbf{A} \circ \mathbf{R}_e$.

To learn embeddings that are invariant to the sensitive attribute and stable against perturbations of the graph structure and non-sensitive attributes, we train the GNN encoder ENC using the Siamese framework [Bromley et al., 1994]. The encoder generates representations $\tilde{\mathbf{z}}_u$ of the augmented graph at every iteration. By generating augmented graphs, NIFTY can induce appropriate bias into the underlying GNN to learn embeddings that are invariant to the combination of counterfactual nodes as well as to random perturbations in the graph structure. A predictor $t : \mathbb{R}^d \rightarrow \mathbb{R}^d$ consisting of a fully-connected neural layer is then used to transform and match the representations with each other. Inspired by Grill et al. [2020], we define a triplet-based objective function that optimizes the similarity between the original graph and its augmented (*i.e.,* counterfactual and noisy) representations:

$$\mathcal{L}_s = \mathbb{E}_u\big[\frac{1}{2}\big(D(t(\mathbf{z}_u), \text{sg}(\tilde{\mathbf{z}}_u)) + D(t(\tilde{\mathbf{z}}_u), \text{sg}(\mathbf{z}_u))\big)\big], \quad (3)$$

where $t(\mathbf{z}_u)$ and $t(\tilde{\mathbf{z}}_u)$ are the transformed representations of node u and perturbed node $\tilde{u}$ respectively, D is the cosine distance, and stopgrad (sg) prevents gradients from being backpropagated. The stopgrad signifies that the node representations $\tilde{\mathbf{z}}_u$ are considered as constant when operating on $t(\mathbf{z}_u)$ and vice-versa.

Finally, the overall objective function for NIFTY is:

$$\min_{\theta_{\text{ENC}}, \theta_t, \theta_f} \; \mathbb{E}_u\left[(1-\lambda)\mathcal{L}_c\right] + \lambda\mathcal{L}_s, \qquad (4)$$

where $\{\theta_{\text{ENC}}, \theta_t, \theta_f\}$ denotes trainable parameters of ENC, predictor t, and classifier f, $\mathcal{L}_c$ is the binary cross entropy (BCE) loss, and the expectation is taken over training nodes in $\mathcal{G}$. The regularization coefficient λ controls the trade-off between downstream node classification loss $\mathcal{L}_c$ and the tripled-based objective $\mathcal{L}_s$. Algorithm 1 summarizes the overall training procedure of NIFTY.

4.2 ENFORCING FAIRNESS AND STABILITY IN GNN ARCHITECTURE

Next, we describe how NIFTY infuses fairness and stability in the architecture of the underlying GNN. In particular, NIFTY modifies the GNN's routing of neural messages. Recall (Sec. 3) that a typical GNN layer is given by:

$\mathbf{h}_u^k = \textsc{Upd}(\textsc{Agg}(\textsc{Msg}(\mathbf{h}_u^{k-1}, \mathbf{h}_v^{k-1}) | v \in \mathcal{N}_u), \mathbf{h}_u^{k-1})$. As we will see in this section, NIFTY modifies the UPD step of each GNN layer.

Without loss of generality, we can consider AGG operator to be a fully-connected layer and UPD to be a non-linear activation function σ. Using these specific parametrizations, the message-passing step can be rewritten as: $\mathbf{h}_u^k = \sigma\big(\mathbf{W}_a^k\ \mathbf{h}_u^{k-1} + \mathbf{W}_n^k \sum_{v \in \mathcal{N}(u)} \mathbf{h}_v^{k-1}\big)$, where $\mathbf{W}_n^k$ is the weight matrix associated with the neighbors of node u at layer k and $\mathbf{W}_a^k$ is the self-attention weight matrix at layer k.

Definition 2 tells us that as the local network neighborhood and the node attribute vector of node u change from $\mathbf{b}_u$ to $\tilde{\mathbf{b}}_u$, the Lipschitz constant L provides an upper bound on how much u's node embedding can change. In fact, the Lipschitz constant L represents the smallest value for which Eqn. 2 in Definition 2 holds true. Leveraging this understanding, NIFTY bounds the change in u's embedding by appropriately normalizing the encoder's weight matrices. This is possible because of the slope-restricted structure of the nonlinear activation function in the UPD step (see proof in Sec. 5). Using our derivations in Sec. 5, at each layer k, we calculate the Lipschitz constant L of term $\mathbf{W}_a^k \mathbf{h}_u^{k-1}$ as the spectral norm of the weight matrix. We use L to normalize $\mathbf{W}_a^k$ as:

$$\tilde{\mathbf{W}}_a^k = \mathbf{W}_a^k / \sigma(\mathbf{W}_a^k). \tag{5}$$

We use this Lipschitz-normalized weight matrix $\tilde{\mathbf{W}}_a^k$ to modify the UPD step as: $\mathbf{h}_u^k = \sigma(\tilde{\mathbf{W}}_a^k\ \mathbf{h}_u^{k-1} + \mathbf{W}_n^k \sum_{v \in \mathcal{N}(u)} \mathbf{h}_v^{k-1})$.

Lipschitz normalization of weight matrices is appealing for two reasons. It bounds the difference between embeddings of original and perturbed nodes (attributes). It also establishes a connection between the stability and counterfactual fairness in a sense that similar inputs should yield similar predictions. Next, we investigate this connection in detail.

Algorithm 1: Overview of NIFTY algorithm

```
Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$; regularization $\lambda$; sensitive
  attribute $s$; number of training epochs num_epoch
Output: Optimized model parameters $\{\theta_{\textsc{Enc}}, \theta_t, \theta_f\}$; fair and
  stable representations $\mathbf{z}_u$ for $u \in \mathcal{G}$
for $ep \leftarrow 1$ to num_epoch do
    for layer $k \leftarrow 1$ to $K$ do
        Lipschitz-normalize ENC's weights $\mathbf{W}_a^k$ (Eqn. 5)
    end
    for node $u \leftarrow 1$ to $|\mathcal{V}|$ do
        Perturb attributes and graph structure to get $\tilde{u}$
          (Sec. 4.1)
        Modify sensitive attribute value to get $\tilde{u}^s$ (Sec. 4.1)
        Encode $\mathbf{z}_u = \textsc{Enc}(u)$, $\tilde{\mathbf{z}}_u = \textsc{Enc}(\tilde{u})$,
          $\tilde{\mathbf{z}}_u^s = \textsc{Enc}(\tilde{u}^s)$
        Transform embeddings: $t(\mathbf{z}_u),\ t(\tilde{\mathbf{z}}_u),\ t(\mathbf{z}_{\tilde{u}}^s)$
          (Sec. 4.1)
    end
    Calculate triplet-based similarity (Eqn. 3)
    Apply downstream classifier $f$ as $\hat{y}_u = f(\textsc{Enc}(u))$
    Update $\{\theta_{\textsc{Enc}}, \theta_t, \theta_f\}$ according to the objective in Eqn. 4
end
```

5 THEORETICAL ANALYSIS OF NIFTY

Here, we provide detailed theoretical analysis of our framework NIFTY. More specifically, we prove that representations generated by NIFTY are stable. We also provide a theoretical upper bound on the unfairness of the resulting representations. Lastly, we show that the downstream classifiers that leverage the representations output by NIFTY satisfy counterfactual fairness as well.

Theorem 1 (NIFTY Stability). *Given a non-linear activation function σ that is Lipschitz continuous, the representations learned by our framework* NIFTY *are stable i.e.,*

$$||\textsc{Enc}(\tilde{u}) - \textsc{Enc}(u)||_p \leq \prod_{k=1}^{K} ||\mathbf{W}_a^k||_p ||(\tilde{\mathbf{b}}_u - \mathbf{b}_u)||_p, \tag{6}$$

where $\tilde{u}$ is a node in the augmented graph which is generated by perturbing the attribute values and/or incident edges of node u, $\mathbf{b}_u$ and $\tilde{\mathbf{b}}_u$ capture all attribute values and incident edge information for nodes u and $\tilde{u}$ respectively, and $\mathbf{W}_a^k$ is weight matrix associated with attributes of node u at layer k.

Proof. Following Sec. 4.2, the node representation output by layer k of the GNN for a perturbed node $\tilde{u}$ is given by:

$$\tilde{\mathbf{h}}_u^k = \sigma\big(\mathbf{W}_a^k\ \tilde{\mathbf{h}}_u^{k-1} + \mathbf{W}_n^k \sum_{v \in \mathcal{N}(\tilde{u})} \mathbf{h}_v^{k-1}\big), \tag{7}$$

where $\mathcal{N}(\tilde{u})$ is the neighborhood of node $\tilde{u}$ which is obtained after perturbing edges incident on node u. Now, the difference between the node embeddings obtained after the message-passing in layer k is:

$$\tilde{\mathbf{h}}_u^k - \mathbf{h}_u^k =$$
$$\sigma\big(\mathbf{W}_a^k \tilde{\mathbf{h}}_u^{k-1} + \mathbf{W}_n^k \sum_{v \in \mathcal{N}(\tilde{u})} \mathbf{h}_v^{k-1}\big) - \sigma\big(\mathbf{W}_a^k \mathbf{h}_u^{k-1} + \mathbf{W}_n^k \sum_{v \in \mathcal{N}(u)} \mathbf{h}_v^{k-1}\big)$$

Taking the norm and assuming that σ is normalized Lipschitz, *i.e.,* $||\sigma(b) - \sigma(a)||_p \leq ||b - a||_p$, we get:

$$||\tilde{\mathbf{h}}_u^k - \mathbf{h}_u^k||_p$$
$$\leq ||\mathbf{W}_a^k(\tilde{\mathbf{h}}_u^{k-1} - \mathbf{h}_u^{k-1}) + \mathbf{W}_n^k(\sum_{v \in \mathcal{N}(\tilde{u})} \mathbf{h}_v^{k-1} - \sum_{v \in \mathcal{N}(u)} \mathbf{h}_v^{k-1})||_p \tag{8}$$

The second term in the above inequality will be close to 0 since the probability of dropping an edge p_e is very small. So, we can drop the second term and then leverage Cauchy-Schwartz inequality to get:

$$\begin{aligned} ||\tilde{\mathbf{h}}_u^k - \mathbf{h}_u^k||_p &\leq ||\mathbf{W}_a^k(\tilde{\mathbf{h}}_u^{k-1} - \mathbf{h}_u^{k-1}) \\ &\leq ||\mathbf{W}_a^k||_p ||(\tilde{\mathbf{b}}_u - \mathbf{b}_u)||_p \end{aligned} \tag{9}$$

Note that the encoder $\textsc{Enc}$ is essentially a sequential composition of message-passing functions applied at layers $1 \cdots K$. Furthermore, the composition of two Lipschitz continuous functions with Lipschitz constants L_1 and L_2 is a new Lipschitz continuous function with $L_1 \times L_2$ as the Lipschitz constant [Gouk et al., 2021]. Putting it all together, we have:

$$||\textsc{Enc}(\tilde{u}) - \textsc{Enc}(u)||_p = ||\tilde{\mathbf{z}}_u - \mathbf{z}_u||_p = ||\tilde{\mathbf{h}}_u^K - \mathbf{h}_u^K||_p \leq \prod_{k=1}^{K} ||\mathbf{W}_a^k||_p ||(\tilde{\mathbf{b}}_u - \mathbf{b}_u)||_p, \quad (10)$$

where K is the last GNN layer. In the case of $p = 2$, the Lipschitz constant in the above equation is equal to the product of the largest singular values (*i.e.,* spectral norm) of weight matrices $\mathbf{W}_a^k$ and can be approximated with a small number of iterations of the power method. We thus perform spectral normalization on the weights of each layer and use the normalized weights $\tilde{\mathbf{W}}_a^k$ in the $\textsc{Upd}$ step of each layer.

Theorem 2 (NIFTY Counterfactual Fairness). *Given a non-linear activation function σ that is Lipschitz continuous and a binary valued sensitive attribute s, the (counterfactual) unfairness of the representations learned by our framework* NIFTY *can be bounded as follows:*

$$||\textsc{Enc}(\tilde{u}^s) - \textsc{Enc}(u)||_p \leq \prod_{k=1}^{K} ||\mathbf{W}_a^k||_p \quad (11)$$

where $\tilde{u}^s$ is a node in the augmented graph which is generated by modifying (flipping) the value of the sensitive attribute (s) of node u while keeping everything else constant.

Proof Sketch. In order to prove this theorem, we will first prove the following:

$$||\textsc{Enc}(\tilde{u}^s) - \textsc{Enc}(u)||_p \leq \prod_{k=1}^{K} ||\mathbf{W}_a^k||_p ||(\tilde{\mathbf{b}}_u^s - \mathbf{b}_u)||_p \quad (12)$$

It can be seen that the above equation has a similar form as that of Eqn. 6 in Theorem 1. Therefore, the above equation can be proved analogously. Note that the node $\tilde{u}^s$ in Eqn. 12 is exactly the same as the node u except that the value of the sensitive attribute is flipped (either from 0 to 1, or from 1 to 0). Therefore, $||(\tilde{\mathbf{b}}_u^s - \mathbf{b}_u)||_p = 1$ and we obtain Eqn. 11.

Proposition 1 (Counterfactual Fairness of Downstream Classifier). *If the representations learned by our framework* NIFTY *satisfy counterfactual fairness, then a downstream classifier $f : \mathbf{z}_u \rightarrow \hat{y}_u$ which leverages these representations also satisfies counterfactual fairness.*

Proof is provided in the Appendix A.

6 EXPERIMENTS

Next, we present experimental results for our NIFTY framework. We address the following key questions: Q1) Does NIFTY enable GNNs to learn fair and stable embeddings? Q2) Can NIFTY achieve group fairness? Q3) How does the interplay between fairness and stability affect downstream performance? Q4) Are changes to GNN's architecture and objective function necessary for fair and stable predictions?

6.1 DATASETS AND EXPERIMENTAL SETUP

We first describe datasets designed to study fair and stable network embeddings and then outline experimental setup.

Datasets. We construct three new datasets. 1) The *German credit graph* has 1,000 nodes representing clients in a German bank that are connected based on the similarity of their credit accounts. The task is to classify clients into good vs. bad credit risks considering clients' gender as the sensitive attribute [Dua and Graff, 2017]. 2) The *Recidivism graph* has 18,876 nodes representing defendants who got released on bail at the U.S state courts during 1990-2009 [Jordan and Freiburger, 2015]. Defendants are connected based on the similarity of past criminal records and demographics. The goal is to classify defendants into bail (*i.e.,* unlikely to commit a violent crime if released) vs. no bail (*i.e.,* likely to commit a violent crime) considering race information as the protected attribute. 3) The *Credit defaulter graph* has 30,000 nodes representing individuals that we connected based on the similarity of their spending and payment patterns [Yeh and Lien, 2009]. The task is to predict whether an individual will default on the credit card payment or not while considering age as the sensitive attribute. See Appendix for details on dataset construction.

Performance evaluation. To measure predictive performance of downstream binary node classification, we use AUROC and F1-score. To quantify group fairness, we use statistical parity (SP) [Dwork et al., 2012], defined as: $\Delta_{SP} = |P(\hat{y}_u=1|s=0) - P(\hat{y}_u=1|s=1)|$, and equal opportunity (EO) [Hardt et al., 2016], defined as: $\Delta_{EO} = |P(\hat{y}_u=1|y_u=1, s=0) - P(\hat{y}_u=1|y_u=1, s=1)|$, where probabilities are estimated on the test set [Dai and Wang, 2021]. To measure counterfactual fairness, we define the unfairness score as the percentage of test nodes for which predicted label changes when the node's sensitive attribute is flipped. Finally, the instability score represents the percentage of test nodes for which predicted label changes when random noise is added to node attributes.

GNN methods. To investigate the flexibility of NIFTY, we incorporate it into five estabished and state-of-the-art GNN methods: GCN [Kipf and Welling, 2017], GraphSAGE [Hamilton et al., 2017], Jumping Knowledge (JK) [Xu et al., 2018], GIN [Xu et al., 2019], and InfoMax [Veličković et al., 2019].

Baseline methods and implementation. We consider two baseline methods: FairGCN [Dai and Wang, 2021] and RobustGCN [Zhu et al., 2019]; all hyperparameters are set

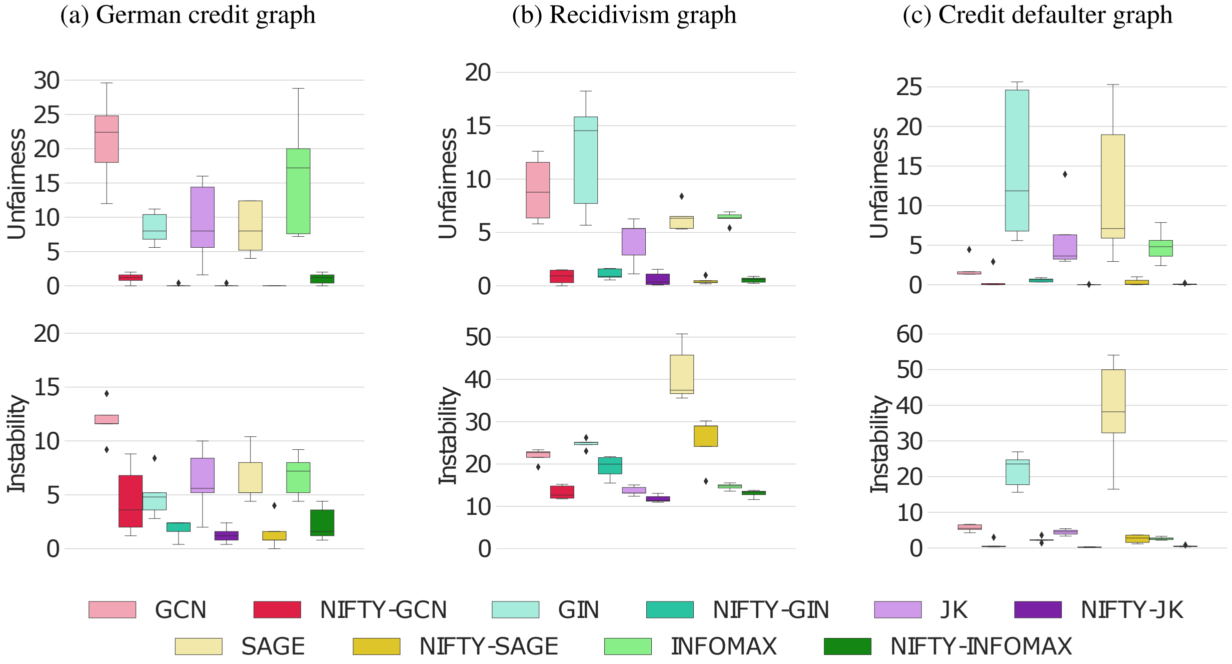


Figure 2: Unfairness (top) and instability (bottom) error rates for five GNNs and their NIFTY counterparts. NIFTY-enhanced GNNs give fairer and more stable predictions than their unmodified counterparts across all three datasets and five GNNs.

Table 1: Comparison of NIFTY to existing methods for improving fairness (*i.e.,* FairGCN [Dai and Wang, 2021]) and stability (*i.e.,* RobustGCN [Zhu et al., 2019]) of GNNs. Shown is average performance across five independent runs. The counterfactual fairness does not apply to FairGCN (*i.e.,* N/A) as FairGCN cannot consider sensitive attributes. Arrows (↑, ↓) indicate the direction of better performance. NIFTY outperforms baselines methods by a large margin.

Dataset	Method	AUROC (↑)	F1-score (↑)	Unfairness (↓)	Instability (↓)	Δ_{SP} (↓)	Δ_{EO} (↓)
German credit graph	FairGCN	75.21±0.36	81.52±0.68	N/A	7.84±2.20	38.12±4.87	26.70±4.27
	RobustGCN	71.06±1.48	78.85±6.39	7.68±4.69	4.48±1.07	25.78±10.92	18.47±9.87
	NIFTY-GCN	70.32±4.42	81.98±0.82	**1.12**±0.77	**4.48**±3.23	**15.08**±8.22	**12.56**±8.60
Recidivism graph	FairGCN	87.55±0.60	78.14±0.94	N/A	24.37±2.33	6.51±0.77	4.51±1.10
	RobustGCN	87.25±1.67	79.02±2.84	2.61±1.58	**13.02**±6.06	5.36±1.28	4.20±1.88
	NIFTY-GCN	81.40±0.89	69.24±0.70	**0.84**±0.68	13.28±1.62	**3.16**±0.60	**2.99**±0.40
Credit defaulter graph	FairGCN	72.69±1.23	80.16±2.03	N/A	5.73±0.60	15.86±5.16	14.43±6.06
	RobustGCN	72.98±0.26	81.79±0.60	0.94±0.60	1.68±0.83	12.41±0.54	10.16±0.49
	NIFTY-GCN	71.92±0.19	81.99±0.63	**0.63**±1.28	**0.95**±1.16	**12.40**±1.62	**10.09**±1.55

following the authors' guidelines. We use stop-gradient operation for training the Siamese networks [Chen and He, 2020]. We set regularization coefficient to $\lambda = 0.6$ in all our experiments and conduct a sensitivity analysis into the effect of λ on NIFTY's performance. See Appendix for details.

6.2 RESULTS

Next, we discuss experimental results that answer key questions highlighted at the beginning of this section (Q1-Q4).

Q1) NIFTY improves fairness and stability of GNNs. Across three datasets and five GNNs, Fig. 2 shows that NIFTY-augmented GNNs learn fairer and more stable embeddings than unmodified GNNs. On average, NIFTY improves stability and fairness of GNNs by 60.87% and 92.01%, respectively. Further, NIFTY can promote fairness and stability of GNNs without sacrificing their predictive performance, as evidenced by AUROC and F1-scores in Table 2. Finally, NIFTY outperforms baseline FairGCN and RobustGCN methods by 62.07% and 57.26% on four fairness and stability metrics (Table 1).

Q2) NIFTY achieves group fairness. Remarkably, while NIFTY's explicit aim is to capture counterfactual fairness, our approach indirectly improves group fairness of GNNs because it reduces information on protected attributes, and, we argue, makes the multi-objective problem of satisfying fairness and stability more tractable. Across three datasets, five GNNs, and two group fairness metrics, NIFTY achieves 43.56% lower Δ_{SP} and 34.70% lower Δ_{EO}. Further, we find that NIFTY achieves 36.05% lower Δ_{SP} and 29.71% lower Δ_{EO} error rates than baseline methods (Table 1), suggesting that in NIFTY, a node's chance of being represented as a particular point in the embedding space does not depend on the node's membership in a protected group.

Table 2: Results of NIFTY for five GNNs and three graph datasets. Shown is average performance across five independent runs. Arrows (↑, ↓) indicate the direction of better performance. NIFTY keeps the predictive power (AUROC and F1-score) of original GNNs while improving their fairness and stability (shaded area).

Dataset	Method	AUROC (↑)	F1-score (↑)	Unfairness (↓)	Instability (↓)	Δ_{SP}(↓)	Δ_{EO}(↓)
German credit graph	GCN	74.00±1.51	80.05±1.20	21.36±6.70	11.84±1.87	41.94±5.52	31.11±4.40
	NIFTY-GCN	70.32±4.42	81.98±0.82	**1.12**±0.77	**4.48**±3.23	**15.08**±8.22	**12.56**±8.60
	GIN	72.69±1.02	82.62±1.55	8.40±2.37	4.96±2.15	14.85±4.64	8.28±6.72
	NIFTY-GIN	69.46±3.99	82.77±0.48	**0.08**±0.18	**1.84**±0.88	**4.39**±3.47	**2.82**±1.60
	GraphSAGE	74.54±0.86	81.15±0.97	8.40±3.93	6.64±2.51	23.79±6.70	15.13±5.74
	NIFTY-GraphSAGE	70.54±2.03	78.14±2.40	**0.00**±0.00	**1.44**±1.54	**6.10**±4.93	**6.34**±3.57
	Infomax	67.98±3.94	72.70±7.91	16.16±9.07	6.80±1.98	36.79±6.58	28.99±5.70
	NIFTY-Infomax	72.01±2.05	81.98±0.33	**1.04**±0.83	**2.32**±1.58	**9.25**±6.45	**7.21**±4.49
	JK	71.49±2.64	80.88±1.02	9.12±6.03	6.24±3.09	20.12±5.16	9.75±4.73
	NIFTY-JK	70.42±2.03	81.25±0.93	**0.08**±0.18	**1.28**±0.77	**4.98**±6.36	**3.42**±3.52
Recidivism graph	GCN	86.52±0.42	77.50±0.87	9.02±3.04	21.97±1.63	8.49±0.73	5.93±0.56
	NIFTY-GCN	81.40±0.89	69.24±0.70	**0.84**±0.68	**13.28**±1.62	**3.16**±0.60	**2.99**±0.40
	GIN	81.32±1.61	70.97±2.48	12.40±5.42	24.82±1.16	9.91±3.24	6.83±3.02
	NIFTY-GIN	84.28±1.42	72.07±6.14	**1.09**±0.49	**19.29**±2.67	**6.57**±1.77	**5.17**±2.15
	GraphSAGE	91.29±0.95	81.58±1.52	6.39±1.24	41.24±6.67	**1.82**±1.51	**2.16**±0.24
	NIFTY-GraphSAGE	92.43±0.44	82.08±2.40	**0.46**±0.32	**25.66**±5.90	6.43±0.67	5.23±1.26
	Infomax	89.24±0.08	80.11±0.16	6.34±0.57	14.69±0.75	7.41±0.48	**3.04**±0.46
	NIFTY-Infomax	79.67±0.44	67.77±1.47	**0.56**±0.27	**13.03**±0.88	**4.04**±0.24	3.43±0.38
	JK	88.60±0.45	79.61±0.82	4.20±2.14	13.64±1.09	7.60±0.71	4.25±0.25
	NIFTY-JK	81.73±0.38	70.20±1.20	**0.64**±0.65	**11.79**±0.88	**4.28**±1.17	**3.65**±1.03
Credit defaulter graph	GCN	72.97±1.63	82.02±0.45	2.04±1.36	5.63±0.98	**10.76**±5.21	**8.71**±4.81
	NIFTY-GCN	71.92±0.19	81.99±0.63	**0.63**±1.28	**0.95**±1.16	12.40±1.62	10.09±1.55
	GIN	73.71±0.33	82.04±0.60	14.89±9.63	21.73±4.81	13.48±2.45	11.19±3.20
	NIFTY-GIN	71.28±0.19	84.97±0.58	**0.59**±0.24	**2.36**±0.78	**4.93**±3.75	**4.60**±2.80
	GraphSAGE	75.19±0.15	82.78±0.37	12.04±9.60	38.19±14.97	15.66±1.62	13.52±1.47
	NIFTY-GraphSAGE	73.27±0.21	83.64±1.66	**0.35**±0.44	**2.57**±1.15	**12.65**±0.95	**9.93**±0.67
	Infomax	74.17±0.11	82.58±0.33	4.87±2.07	2.67±0.43	14.57±0.69	12.26±0.72
	NIFTY-Infomax	71.86±0.26	81.70±0.06	**0.09**±0.08	**0.53**±0.20	**11.83**±0.36	**9.52**±0.31
	JK	73.80±0.06	82.70±0.73	6.03±4.63	4.45±0.83	12.70±1.74	9.51±0.07
	NIFTY-JK	72.07±0.30	81.78±0.08	**0.02**±0.02	**0.26**±0.09	**11.77**±0.09	**9.42**±0.37

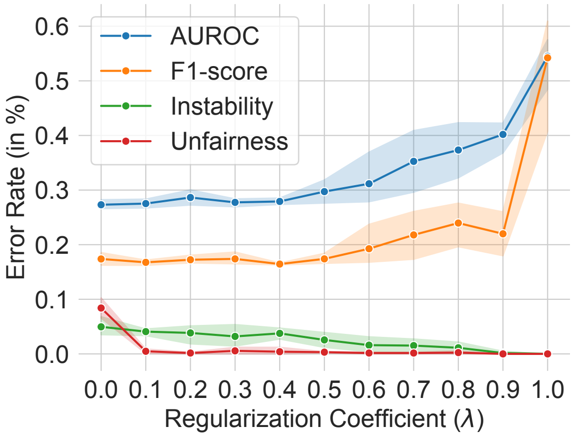


Figure 3: The effects of regularization on the performance of NIFTY. Shown are results for NIFTY-GIN and the German credit graph (see Fig. 4 for other datasets). Over a wide range of regularization strength ($0.1 < \lambda < 0.5$), NIFTY achieves a near-perfect stability and fairness on the downstream task without sacrificing the predictive ability of GIN.

Q3) Trade-offs between fairness, stability, and predictive performance. As we increase regularization coefficient λ in NIFTY (Fig. 3), we find that the error rates for counterfactual fairness and stability steadily decrease. Interestingly, even with a modest amount of regularization ($\lambda = 0.1$), NIFTY achieves a 94.29% improvement in unfairness error rate. As expected, a more strongly regularized NIFTY model takes a hit on its predictive performance (higher error rate for AUROC and F1-score). See Fig. 4 for similar trends on the recidivism and credit defaulter graphs.

Q4) Ablation study. We conduct ablations on two key NIFTY's components, namely the objective function and the layer-wise normalization of GNN's architecture using the Lipschitz constant. Results show that both components are necessary to generate embeddings that are simultaneously fair and stable (Table 3). In particular, we observe a 90.7% improvement in fairness of NIFTY-GCN as compared to vanilla GCN, providing empirical evidence for our theoretical analysis that the Lipschitz normalization can improve both fairness and stability of graph embeddings (Section 5).

Table 3: Ablation study on the recidivism graph. Shown is average performance across five independent runs, evidencing that NIFTY's changes in the GNN architecture and the objective function are complementary and improve fairness and stability.

Method	AUROC (↑)	F1-score (↑)	Unfairness (↓)	Instability (↓)	$\Delta_{SP}(\downarrow)$	$\Delta_{EO}(\downarrow)$
GCN [Kipf and Welling, 2017]	86.52±0.42	77.50±0.87	9.02±3.04	21.97±1.63	8.49±0.73	5.93±0.56
NIFTY-GCN w/o obj. changes (Sec. 4.1)	80.02 ±0.20	67.51 ±0.23	2.61±0.64	13.69±0.60	5.86±0.85	4.65±0.49
NIFTY-GCN w/o arch. changes (Sec. 4.2)	84.83 ±2.85	76.15±5.74	1.64 ±1.58	13.98 ±1.38	4.29 ±1.32	3.48 ±1.37
NIFTY-GCN	81.40 ±0.89	69.24±0.70	**0.84**±0.68	**13.28**±1.62	**3.16**±0.60	**2.99**±0.40

7 CONCLUSIONS & FUTURE WORK

We propose and address the problem of learning representations that are both fair and stable. To this end, we introduce NIFTY, a unified framework which exploits a key connection between counterfactual fairness and stability to learn representations that satisfy both these properties. At its core, NIFTY, outlines a two-level strategy to modify an existing GNN both at the architectural as well as the objective function level. We carry out detailed theoretical analysis to show that the representations learned by NIFTY are both counterfactually fair and stable. Further, results on new graph datasets from domains such as criminal justice and financial lending show that NIFTY can considerably improve fairness (both in terms of counterfactual and group fairness) and stability without sacrificing predictive performance. This work paves way for several exciting future directions. For instance, it would be interesting to extend NIFTY to generate fair and stable representations of other graph components (*e.g.,* edges, subgraphs) and to cater to other downstream tasks (*e.g.,* link prediction, graph classification).

Acknowledgements

We would like to thank the anonymous reviewers for their insightful feedback. H.L. is supported, in part, by the NSF award IIS-2008461, and Google. M.Z. is supported, in part, by NSF under nos. IIS-2030459 and IIS-2033384, the Harvard Data Science Initiative, the Amazon Research Award, and the Bayer Early Excellence in Science Award. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies.

References

Emily Alsentzer, Samuel G Finlayson, Michelle M Li, and Marinka Zitnik. Subgraph neural networks. In *NeurIPS*, 2020.

Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. In *Sociological Methods & Research*, 2018.

Avishek Joey Bose and William L Hamilton. Compositional fairness constraints for graph embeddings. In *ICML*, 2019.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *NeurIPS*, 1994.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv*, 2020.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. In *Big Data*, 2017.

Enyan Dai and Suhang Wang. Fairgnn: Eliminating the discrimination in graph neural networks with limited sensitive attribute information. In *WSDM*, 2021.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *ITCS*, 2012.

Joseph Fisher, Arpit Mittal, Dave Palfrey, and Christos Christodoulopoulos. Debiasing knowledge graph embeddings. In *EMNLP*, 2020.

Pablo Gainza, Freyr Sverrisson, Frederico Monti, Emanuele Rodola, D Boscaini, MM Bronstein, and BE Correia. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. In *Nature Methods*, 2020.

Simon Geisler, Daniel Zügner, and Stephan Günnemann. Reliable graph neural networks via robust aggregation. In *NeurIPS*, 2020.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. In *Machine Learning*. Springer, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

Deisy Morselli Gysi, Ítalo Do Valle, Marinka Zitnik, Asher Ameli, Xiao Gan, Onur Varol, Helia Sanchez, Rebecca Marlene Baron, Dina Ghiassian, Joseph Loscalzo, et al. Network medicine framework for identifying drug repurposing opportunities for COVID-19. *arXiv*, 2020.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. In *Applied and Computational Harmonic Analysis*, 2011.

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *NeurIPS*, 2016.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *WWW*, 2020.

Kexin Huang, Cao Xiao, Lucas M Glass, Marinka Zitnik, and Jimeng Sun. Skipgnn: predicting molecular interactions with skip-graph networks. In *Scientific Reports*, 2020.

Guangyin Jin, Qi Wang, Cunchao Zhu, Yanghe Feng, Jincai Huang, and Jiangping Zhou. Addressing crime situation forecasting task with temporal graph convolutional neural network approach. In *ICMTMA*, 2020.

Kareem L Jordan and Tina L Freiburger. The effect of race/ethnicity on sentencing: Examining sentence type, jail length, and prison length. In *Journal of Ethnicity in Criminal Justice*. Taylor & Francis, 2015.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In *ITCS*, 2017.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. In *NeurIPS*, 2017.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *ICML*, 2019.

Jiachun Liao, Chong Huang, Peter Kairouz, and Lalitha Sankar. Learning generative adversarial representations (gap) under fairness and censoring constraints. *arXiv*, 2019.

Tahleen A Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. Fairwalk: Towards fair graph embedding. In *IJCAI*, 2019.

Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *FAT*, 2019.

Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *ICLR*, 2019.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. In *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *ICML*, 2018.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

I-Cheng Yeh and Che-hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. In *ESA*, 2009.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *PKDD*, 2018.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *WWW*, 2017a.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, Krishna P Gummadi, and Adrian Weller. From parity to preference-based notions of fairness in classification. *arXiv*, 2017b.

Xiang Zhang and Marinka Zitnik. GNNguard: Defending graph neural networks against adversarial attacks. In *NeurIPS*, 2020.

Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *KDD*, 2019.

Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. In *Bioinformatics*, 2018.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *ICLR*, 2019.

A PROPOSITION 1 AND ITS PROOF

Proposition 1 (Counterfactual Fairness of Downstream Classifier). *If the representations learned by our framework* NIFTY *satisfy counterfactual fairness, then a downstream classifier $f : \mathbf{z}_u \rightarrow \hat{y}_u$ which leverages these representations also satisfies counterfactual fairness.*

Proof. The downstream classifier uses the representation $\mathbf{z}_u$ output by our framework for predicting the label $\hat{y}_u$ of node u, thus forming a Markov chain $\mathbf{x}_u \rightarrow \mathbf{z}_u \rightarrow \hat{y}_u$ [Liao et al., 2019]. As we discuss in Section 3, node representations are said to be counterfactually fair if they are independent of the sensitive attribute. *i.e.,* the mutual information between the sensitive attribute s and the representation $\mathbf{z}_u$ for any given node u is zero: $I(s; \mathbf{z}_u) = 0$.

Using the properties of inequality and non-negativity of mutual information:

$$0 \leq I(s; \hat{y}_u) \leq I(s; \mathbf{z}_u) \text{ and } I(s; \mathbf{z}_u)=0 \implies I(s; \hat{y}_u)=0 \tag{13}$$

Therefore, the node label $\hat{y}_u$ for any given node u is independent of the sensitive attribute s, and consequently the downstream node classifier satisfies counterfactual fairness.

B DATASET DETAILS

German Credit Graph. The German Graph credit dataset classifies people described by a set of attributes as good or bad credit risks [Dua and Graff, 2017]. It consists of attributes like Gender, LoanAmount, and other account-related features of 1,000 clients. We use Minkowski distance as the similarity measure for calculating the similarity between two node attributes using: $1/(1 + \text{minkowski}(\mathbf{x}_u, \mathbf{x}_v))$. To obtain the credit graph network that connects clients, we connect two nodes if the similarity between them is 80% of the maximum similarity between all respective nodes (Refer Table. 4 for details). We argue that a graph neural network is fair if it predicts the client credit risk irrespective of their gender. Hence, we used *gender* as the sensitive attribute for the loan dataset.

Recidivism Graph. The dataset consists of samples of bail outcomes collected from several state courts in the US between 1990-2009 [Jordan and Freiburger, 2015]. It consists of past criminal records, demographic attributes, and other details of 18,876 defendants who got released on bail. We use Minkowski distance as the similarity measure for calculating the similarity between two node attributes using: $1/(1 + \text{minkowski}(\mathbf{x}_u, \mathbf{x}_v))$. To obtain the bail graph network that connects defendants, we connect two nodes if the similarity between them is 60% of the maximum similarity between all respective nodes (Refer Table. 4 for details). A machine learning model is trained to predict a defendant who is more likely to commit a violent or nonviolent crime once released on bail. A fair model should make predictions independent of the defendant's *race*, and, thus, we use it as the protected attribute for the dataset.

Credit Defaulter Graph. We use a processed version [Ustun et al., 2019] of the credit dataset in Yeh and Lien [2009]. The task is to predict whether an applicant will default on an upcoming credit card payment. The dataset contains 30,000 individuals with features like education, credit history, age, and features derived from their spending and payment patterns. We use Minkowski distance as the similarity measure for calculating the similarity between two node attributes using: $1/(1 + \text{minkowski}(\mathbf{x}_u, \mathbf{x}_v))$. To obtain the credit defaulter graph network that connects applicants, we connect two nodes if the similarity between them is 70% of the maximum similarity between all respective nodes (Refer Table. 4 for details). For the credit dataset, we used *age* as the sensitive attribute.

C ARCHITECTURE AND HYPERPARAMETER SELECTION

We provide an overview of the important components of our proposed architecture and their respective training settings.

Encoder. The encoder block of our proposed framework can comprise of either simple Multilayer Perceptron (MLP) networks or any other GNN variant. For all our experiments, we use the vanilla GNN as the encoder block of our contrastive learning framework. For all datasets, we use a single-layer GNN encoder and set the hidden dimensionality to 16. The encoder is followed by a two-layer MLP projection head [Chen et al., 2020]. We only use ReLU and BatchNormalization (BN) layers after the first hidden layer in the MLP. For both the MLP layers, we set the hidden dimensionality to 16.

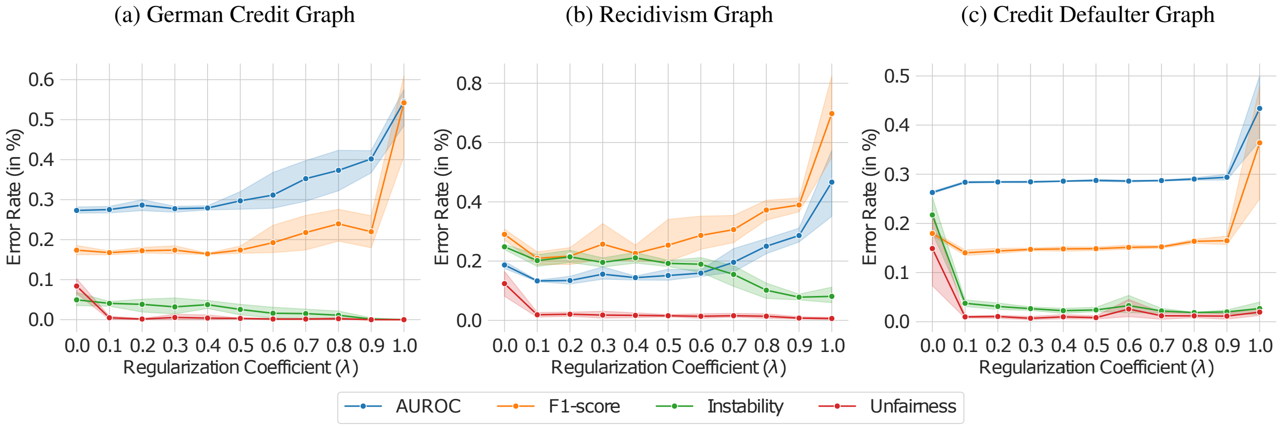
Predictor. We use a single layer MLP with no ReLU and BN as our predictor [Chen et al., 2020] to transform the graph embeddings of one augmented graph to another and vice-versa. We set the hidden dimensionality to 16 for the predictor layer.

Downstream classifier. We use a single fully-connected layer with a Sigmoid activation function in all our node-classification experiments. We set the hidden dimensionality of the fully-connected layer to 16.

Hyperparameters. For all experiments, we set the probability of perturbing a feature dimension to $p_n = 0.1$ and the probability with which an edge is dropped to $p_e = 0.001$. For training GNNs and their NIFTY-augmented counterparts (Sec. 6.1), we use an Adam optimizer with a learning rate of 1×10^{-3}, weight decay of 1×10^{-5}, and the number of epochs to 1000. For RobustGCN and FairGCN, all hyperparameters are set following the authors' guidelines.

Table 4: Statistics of novel graph datasets designed for node classification and accompanied by sensitive attributes. The datasets are appropriate to study fairness- and stability-aware algorithms.

Dataset	**German credit graph**	**Recidivism graph**	**Credit defaulter graph**
Nodes	1,000	18,876	30,000
Edges	22,242	321,308	1,436,858
Node features	27	18	13
Average node degree	44.48$\pm$26.51	34.04$\pm$46.65	95.79$\pm$85.88
Sensitive attribute	Gender (Male/Female)	Race (Black/White)	Age ($\leq 25/>25$)
Node labels	good credit vs. bad credit	bail vs. no bail	payment default vs. no default


(a) German Credit Graph

(b) Recidivism Graph

(c) Credit Defaulter Graph

Figure 4: Effect of regularization coefficient on AUROC, F1-score, stability, and fairness in NIFTY-GIN on (a) the German credit graph, (b) the recidivism graph, and (c) the credit defaulter graph. With increasing the regularization coefficient on the self-supervised task the robustness and fairness score can reach 0% error.